Subject: Section 1.4 from 03-107r1, revised
From: Van Snyder

1 Edits for Section 5

Edits refer to 02-007r3. Page and line numbers are displayed in the margin. Absent other instructions, a page and line number or line number range implies all of the indicated text is to be replaced by associated text, while a page and line number followed by + (-) indicates that associated text is to be inserted after (before) the indicated line. Remarks are noted in the margin, or appear between [and] in the text.

Most of the following changes were proposed in paper 02-279r2 at J3 meeting 163.

[Note 5.3 contradicts 12.1.2.3, where it says “a dummy procedure with the pointer attribute is a dummy procedure pointer.” It’s also hard to imagine how a dummy procedure can avoid being a dummy argument. Editor: Delete “a dummy argument that is” at [69:1], insert “without the POINTER attribute” after “procedure” at [69:2], and delete Note 5.3.] 69:1-2, Note 5.3

C529a (R501) The VALUE attribute shall not be specified for a dummy procedure. 69:30+

The following change was proposed at J3 meeting 163 in paper 02-305r1, and part 4 of paper 02-319r2.

5.1.1.7 TYPE

73:16-26

A TYPE type specifier may be used to declare entities of a derived type.

If the *derived-type-spec* contains a *type-name*, then the TYPE type specifier is used to declare entities of the derived type specified by that *type-name*. Derived type parameter values for each such entity may be specified by a *type-param-spec-list* in the *derived-type-spec*. The components of each such entity are declared to be of the types specified by the corresponding *component-def-stmt*s of the *derived-type-def* (4.5.1).

Where a data entity is declared explicitly using the TYPE type specifier, the specified derived type shall have been defined previously in the scoping unit or be accessible there by use or host association. If the data entity is a function result, the derived type may be specified in the FUNCTION statement provided the derived type is defined within the body of the function or is accessible there by use or host association. If the derived type or is specified in the FUNCTION statement and is defined within the body of the function, it is as if the function result variable was declared with that derived type immediately following the *derived-type-def* of the specified derived type.

A scalar entity of derived type is a **structure**. If a derived type has the SEQUENCE property, a scalar entity of the type is a **sequence structure**.

5.1.1.8 CLASS

[Every other reference to “use association” has “use” in lower case. Editor: “USE” ⇒ “use”.] 74:31

[Editor: Insert a comma before “ then” (only one other “then” in Section 5 doesn’t have one).] 75:13

[What does “Nonkind type parameters can be deferred” do here? Nothing. Editor: Delete the sentence.] 77:19-20

[What does “Nonkind type parameters can be deferred” do here? Nothing. Editor: Delete the sentence.] 77:23

[Editor: Insert “abstract” before “interface”.] 78:35

[Editor: Insert a comma before “ then” (only one other “then” in Section 5 doesn’t have one).] 83:7

The next three changes were proposed in paper 02-316r1 at J3 meeting 163.

[Editor: After “association”, add “and to the target when it is accessed through the pointer”.] 83:10

[Editor: Delete Note 5.25.] 83:11-

[Editor: Remove “and”. After “status”, add “and value”.] 83:12

[Editor: There are two consecutive index items with a space between them at around line 1447 in 85:26

c05.tex. This causes an extra blank between “the” and “SAVE”.]

[The word “become” implies a dynamic event that occurs at some instant during the execution sequence. Storage association is a static condition. Editor: replace “become” with “be” at the following places: [98:1], [98:3], [98:6], [98:10], [98:12], and [98:17].] 98:1-17